Multiagent and Grid Systems – An International Journal 4 (2008) 213–233
IOS Press

Automated formal analysis of human multi-issue negotiation processes

Tibor Bosse[a,*], Catholijn M. Jonker[b], Lourens van der Meij[a] and Jan Treur[a]
[a]*Vrije Universiteit Amsterdam, Department of Artificial Intelligence, De Boelelaan 1081a, 1081 HV Amsterdam, The Netherlands*
[b]*Delft University of Technology, Man Machine Interaction Group, Department of Mediametics, Mekelweg 4, 2628 CD, Delft, The Netherlands*

Abstract. This paper reports on experiments in (human) multi-issue negotiation and their analysis, and presents a generic software environment supporting such an analysis. First, the paper presents a System for Analysis of Multi-Issue Negotiation (SAMIN). SAMIN was designed to analyse multi-issue negotiation processes between human negotiators, between human and software agents, and between software agents. The agents conduct one-to-one negotiations, in which the values across multiple issues are negotiated on simultaneously. To analyse such negotiation processes, the user can enter any formal property deemed useful into the system and use the system to automatically check this property in given negotiation traces. The paper presents the results of applying SAMIN in the analysis of empirical traces obtained from an experiment in multi-issue negotiation about second hand cars. In the experiment the efforts of 74 humans negotiating against each other have been analysed using SAMIN.

Keywords: Multi-issue negotiation, human, formal analysis, tool

1. Introduction

Negotiation is a process by which a joint decision is made by two or more parties [19]. Typically each party starts a negotiation by offering the most preferred solution from the individual area of interest. If an offer is not acceptable by the other parties they make counter-offers in order to move them closer to an agreement. The field of negotiation can be split into different categories, e.g. along the following lines:

- one-to-one versus more than two parties
- single- versus multi-issues
- closed versus open
- mediator-based versus mediator-free

The research reported in this article concerns one-to-one, multi-issue, closed, mediator-free negotiation. For more information on negotiations between more than two parties (e.g., in auctions), the reader is referred to, e.g. [23]. In single-issue negotiation, the negotiation focuses on one aspect only (typically price) of the concept under negotiation. Multi-issue negotiation (also called multi-attribute negotiation) is often seen as a more cooperative form of negotiation, since often an outcome exists that brings joint gains for both parties, see [21].

*Corresponding author. E-mail: tbosse@cs.vu.nl.

ISSN 1574-1702/08/$17.00

Closed negotiation means that no information regarding preferences is exchanged between the negotiators. The only information exchanged is formed by the bids. In partially open negotiation some information regarding preferences is exchanged, and in completely open negotiation all information is exchanged. More information about (partially) open negotiations can be found, e.g., in [17,21]. However, the trust necessary for open negotiations is not always available.

The use of mediators is a well-recognised tool to help the parties in their negotiations, see e.g. [6,10]. The mediator aims for a deal that is fair to all parties. Reasons for negotiating without a mediator can be the lack of a trusted mediator, the costs of a mediator, and the hope of doing better than fair with respect to personal gain.

The literature on closed, multi-issue, one-to-one negotiation without mediators covers both systems to (partially) automate the negotiation process, and more analytic research focused on properties of the negotiation process and negotiation space. Based on a literature study and on further analysis, a number of properties are presented here that focus largely on the dynamics of the negotiation process itself and on the results of the negotiation.

The SAMIN system presented in this paper has been developed to support and formally analyse such negotiation processes, i.e., multi-issue, (partially) closed, one-to-one negotiations without mediators. The system requires three types of input:

1. a negotiation *trace* (or a set of traces)
2. a set of *dynamic properties* considered relevant for the negotiation process
3. the *negotiation profiles* of the participants

A *trace* is a sequence of bids by the negotiators. A *dynamic property* is an (informal, semi-formal or formal) expression that might or might not hold for a certain trace. An example of a simple dynamic property is bid-alternation, i.e., after communicating a bid to another agent, the agent remains silent until it has received a new bid from the other agent. A *negotiation profile* is a description of the preferences of the agent within the particular negotiation domain. The profiles together define the space of possible and efficient outcomes and are, therefore, essential for the creation of a complete analysis of the performance of a negotiator.

The most important measure of efficiency in bilateral negotiations, cf. [20], is Pareto-efficiency. An outcome is said to be Pareto-efficient if the utility of any party cannot be improved without a loss of utility for another. The set of all Pareto-efficient outcomes form the Pareto-efficient frontier. The distance of an outcome to the Pareto frontier gives a measure of efficiency of a bid.

The SAMIN system consists of three components: an *Acquisition Component*, an *Analysis Component* and a *Presentation Component*. The Acquisition Component is used to acquire the input necessary for analysis. The Analysis Component performs the actual analysis, and the Presentation Component presents the results of the analysis in a user-friendly format.

SAMIN can check automatically whether selected properties hold for the traces under analysis. Such an analysis provides a means to improve bidding strategies and bidding protocols, both for human negotiators and for software agents in automated negotiation systems. Beside introduction of the SAMIN system, a subgoal of this paper is to report some results of such analyses, focusing on human negotiators. The results are presented of applying SAMIN in the analysis of empirical traces obtained from an experiment in multi-issue negotiation about second hand cars. In the experiment the efforts of 74 humans negotiating against each other have been analysed using SAMIN.

In Section 2 formalisation of negotiation process dynamics will be discussed in terms of negotiation states, transitions, and traces. Section 3 explains the formal specification of dynamic properties and presents example dynamic properties relevant for (partially) closed multi-issue one-to-one negotiations. The architecture of the SAMIN system is presented in Section 4. Section 5 illustrates how SAMIN can be used to analyse human negotiation processes. Some experiments in human multi-issue negotiation are described and analysed, and the results of the analysis are discussed. Finally, Section 6 discusses related work, and Section 7 provides conclusions and some planned future work.

2. Formalising negotiation process dynamics

Negotiation is essentially a dynamic process. To analyse those dynamics, it is, therefore, relevant to formalise and study dynamic properties of such processes. For example, how does a bid at a certain point in time compare to bids at previous time points? The formalisation introduced in this section is based on the notion of negotiation process state, negotiation transition and negotiation trace.

2.1. Formalising states of a negotiation process

The state of a (one-to-one) negotiation process at a certain time point can be described as a combined state consisting of two states for each of the negotiating agents:

$$\mathsf{S} = < \mathsf{S1}, \mathsf{S2} >$$

where S1 is the state of agent A, and S2 is the state of agent B.

Each of these states includes:

– the agent's own most recent bid
– its evaluation of its own most recent bid
– its evaluation of the other agent's most recent bid
– the history of bids from both sides and evaluations

To describe negotiation states a state ontology Ont is used. Example elements of this ontology are a sort BID for bids, and relations such as utility(A, b, v) expressing that A's overall evaluation of bid b is a real number v between 0 and 1. Based on this ontology the set of ground atoms At(Ont) can be defined. A state is formalised as any truth assignments: At(Ont) $\rightarrow \{t, f\}$ to this set of ground atoms. The set of all states described by this ontology is denoted by States(Ont).

2.2. Negotiation transitions

A particular negotiation process shows a sequence of transitions from one state S from States(Ont) to another (next) state S' from States(Ont). A transition S $\rightarrow$ S' from a state S to S' can be classified according to which agents are involved. During such a transition each of the main state components (S1, S2) of the overall state S may change. The simplest types of transition involve a single component transition. For example, when one agent generates a bid, while the other agent is just waiting: a transition of type S1 $\rightarrow$ S1 or S2 $\rightarrow$ S2 occurs. Next come transition types in which both components are involved. For example, when a communication between agent A and agent B takes place, changing the state S2 of agent B, a transition of type S1 x S2 $\rightarrow$ S2 occurs. Notice that in principle, also more complex transition types are possible, involving changes of both state components at the same time, i.e., S1 x S2 $\rightarrow$ S1 x S2. In organised cooperations between multiple agents the complexity of the types of transitions is often limited by regulation of the organisation. For example, in organised negotiation processes, usually it is assumed in the protocol that after communicating a bid to the other agent, the agent remains silent until it has received a new bid from the other agent (see the dynamic property 'bid alternation' in Sections 3 and further below). Such an assumption about the protocol implies that the transitions involved in the negotiation are only of the simpler types mentioned above.

2.3. Negotiation traces

Negotiation traces are time-indexed sequences of negotiation states, where each successive pair of states is a negotiation transition. To describe such sequences a fixed *time frame* T is assumed which is linearly ordered. A *trace* $\mathcal{T}$ over a state ontology Ont and time frame T is a mapping $\mathcal{T}$: T $\rightarrow$ STATES(Ont), i.e., a sequence of states T_t (t $\in$ T) in STATES(Ont). The set of all traces over state ontology Ont is denoted by TRACES(Ont). Depending on the application, the time frame T may be dense (e.g., the real numbers), or discrete (e.g., the set of integers or natural numbers or a finite initial segment of the natural numbers), or any other form, as long as it has a linear ordering.

3. Dynamic properties of negotiation processes

This section presents a classification of dynamic properties of negotiation processes along with examples of each class. Before presenting the classification and the specific dynamic properties of negotiation, the formal method for specifying those properties is presented.

3.1. Specification of dynamic properties

Specification of dynamic properties of a negotiation process can be done in order to *analyse* its dynamics, for example to find out how certain properties of a negotiation process as a whole relate to properties of a certain subprocess, or to verify or evaluate a negotiation model. To formally specify dynamic properties that express characteristics of dynamic processes (such as negotiation) from a temporal perspective an expressive language is needed. To this end the *Temporal Trace Language* TTL is used as a tool; cf. [5]. This language can be classified as a reified predicate-logic-based temporal logic; see, e.g. [10,11], in contrast to, for example, modal-logic-based temporal logics as the ones discussed in, e.g. [9]. The language is briefly introduced here. For more details, see [1].

The set of dynamic properties DYNPROP(Ont) is the set of temporal statements that can be formulated with respect to traces based on the state ontology Ont in the following manner. Given a trace $\mathcal{T}$ over state ontology Ont, a certain state of the agent A during a negotiation process at time point t is indicated by state($\mathcal{T}$, t, A). In the third argument, instead of A also specific parts of A can be used, such as input(A), or output(A), which refer to observations and actions by A, or communications. These state indicators can be related to state properties via the formally defined satisfaction relation denoted by the infix predicate $|=$, comparable to the Holds-predicate in the Situation Calculus: state($\mathcal{T}$, t, A) $|=$ p denotes that state property p holds in trace $\mathcal{T}$ at time t in the state of agent A. Here state properties are considered objects and denoted by term expressions in the TTL language. The infix predicate $|=$ is alternatively written in prefix form as holds: holds(s, p) means s $|=$ p. Based on these atomic TTL-statements, dynamic properties can be formulated in a formal manner in a sorted predicate logic with sorts TIME for time points, STATE for states, Traces for traces and STATPROP for state formulae, using quantifiers and the usual logical connectives such as $\neg$, &, $\vee$, $\Rightarrow$, $\forall$, $\exists$.

In some more detail, both state (γ, t, output(A)) and p are terms of the TTL language. TTL terms are constructed by induction in a standard way for sorted predicate logic from variables, constants and function symbols typed with TTL sorts. Dynamic properties are expressed by TTL-formulae inductively defined by:

1. If v_1 is a term of sort STATE, and u_1 is a term of the sort STATPROP, then holds(v_1, u_1) is an atomic TTL formula.
2. If τ_1, τ_2 are terms of any TTL sort, then $\tau_1 = \tau_2$ is an atomic TTL formula.
3. If t_1, t_2 are terms of sort TIME, then $t_1 < t_2$ is an atomic TTL formula.
4. The set of well-formed TTL-formulae is defined inductively in a standard way based on atomic TTL-formulae using Boolean connectives and quantifiers.

For example, the dynamic property
'in any trace γ, if at any point in time t1 agent A observes that it is dark in the room, whereas at an earlier time point t0 it observed that a light was on in this room, then there exists a point in time t2 after t1 such that at t2 in the trace γ agent A switches the lamp on again' is expressed in formalized form in TTL as:

$\forall t1$ [[state(γ, t1, input(A)) $|=$ observed(dark_in_room) &
$\exists t0 < t1$ [state(γ, t0, input(A)) $|=$ observed(light_on)]
$\Rightarrow$ $\exists t2 \geqslant t1$ state(γ, t2, output(A)) $|=$ performing_action(switch_on_light)]

As another example, consider the dynamic property bid alternation, which states that for all two different moments in time t1, t3, that A generates a bid, there is a moment in time t2, with t1 $<$ t2 $<$ t3, such that A received a bid generated by B. In formal TTL-format, this property is expressed as:

bid_alternation(γ**:TRACE**) $\equiv$
$\forall$ A, B: AGENT, $\forall$ b1, b3: BID, $\forall$ t1, t3: time :
t1 $<$ t3 &
state(γ, t1, output(A)) $|=$ to_be_communicated_to_by(b1, B, A) &
state(γ, t3, output(A)) $|=$ to_be_communicated_to_by(b3, B, A)
$\Rightarrow$ $\exists$b2: BID, $\exists$t2: time : t1 $<$ t2 $<$ t3 &
state(γ, t2, input(A)) $|=$ communicated_to_by(b2, A, B)

Often for reasons of presentation dynamic properties are expressed in informal or semi-formal forms.

3.2. Classes and examples of dynamic properties of negotiation

The properties relevant for analysing the dynamics of (partially) closed multi-issue one-to-one negotiation, can be divided into the following types:

– **Bid properties** give some information about a specific bid. They are usually defined in terms of the negotiation space and the profiles of the negotiators. Bid properties concern, for example, the Pareto efficiency of a bid.
– **Result properties** are a subset of the set of bid properties, concerning only the last bid of a negotiation process (i.e., the final agreement).

- **Bid comparison properties** compare two arbitrary bids with each other. An example is domination: a bid b1 dominates a bid b2 with respect to agents A and B iff both agents prefer bid b1 over bid b2; see below for a formalisation
- **Step properties** are a subset of the set of bid comparison properties, concerning only the transitions between successive bids. Hence, they are restricted to the combinations of bids of one party that directly follow each other.
- **Limited interval properties** concern parts of traces. Basically, they state that each step in a certain interval satisfies a certain step property. For instance: a negotiation process is Pareto-monotonous for the interval [t1, t2] iff for all successive bids b1, b2 in the interval b2 dominates b1 (see below).
- **Trace properties** are a subset of the set of limited interval properties, concerning whole traces.
- **Multi-trace properties** compare the dynamics observed in more than one trace. An example is Better Negotiator: agent A is a better negotiator than agent B iff in more than 60% of the negotiations between A and B, the deal reached is more to the advantage of agent A than of agent B.
- **Protocol properties** specify certain constraints on the negotiation protocol. A specific instance is: over time the bids of negotiators A and B alternate.

Note that the first two types are basically *static properties*, whereas the other types are *dynamic properties*: they specify behaviour over time. In [3] for each of these types a number of properties are described in detail, both in informal and in formal notation. In this paper, only a small selection of relevant properties is presented. See Appendix A for a more extensive overview of these properties.

configuration_differs(b1:BID, b2:BID) $\equiv$
$\exists$a: ISSUE, $\exists$v1, v2: VALUE:
value_of(b1, a, v1) & value_of(b2, a, v2) & v1 $\neq$ v2

This bid comparison property states that two bids b1 and b2 differ in configuration iff there is an issue that has a different value in both bids. For example, in bid b1 the value of the issue "color" is "red", whereas in bid b2 this value is "blue". Similar properties can be defined stating that two bids differ in configuration in at least x issues. This property can also be used as a building block to specify a step property, e.g. "in the view of agent A, agent B varies the configuration, but not the utility". Such a property are useful to find out what kind of opponent the negotiator is dealing with.

strictly_dominates(b1:BID, b2:BID, A:AGENT, B:AGENT) $\equiv$
$\forall$vA1, vA2, vB1, vB2: real:
util(A, b1, vA1) & util(A, b2, vA2) &
util(B, b1, vB1) & util(B, b2, vB2) $\Rightarrow$ vA1 $>$ vA2 & vB1 $>$ vB2

This bid comparison property states that a bid b1 dominates a bid b2 with respect to agents A and B iff both agents prefer bid b1 over bid b2. This notion is related to Pareto Efficiency, see e.g. [21]. The property could also be changed to weakly dominates by changing the $>$ sign into the $\geqslant$ sign. Moreover, it can be used as a building block to specify step properties, limited interval properties (see the next property), and trace properties.

strict_pareto_monotony(γ:trace, tb:time, te:time) $\equiv$
$\forall$t1, t2: time, $\forall$A, B: AGENT, $\forall$b1, b2: BID:
[tb $\leqslant$ t1 $<$ t2 $\leqslant$ te & is_followed_by(γ, A, t1, b1, B, t2, b2)]

$\Rightarrow$ state(γ, t2) $|=$ strictly_dominates(b2, b1, A, B)

This limited interval property makes use of the previous property. It states that a negotiation process γ is strictly Pareto-monotonous for the interval [t1, t2] iff for all successive bids b1, b2 in the interval b2 dominates b1. By choosing for tb and te respectively the start and end time of the process, the property can be transformed into a trace property. Generally, traces that satisfy this property are not abundant in (human) real world multi-issue negotiations, since if the profiles of the two parties are strongly opposed (with emphasis on the same issues), even in multi-issue situations a gain for the one often implies a loss for the other. If, however, the profiles are less opposed, Pareto-monotony may occur.

pareto_inefficiency(γ:trace b:BID, A:AGENT, B:AGENT, ε:real) $\equiv$
$\forall$vA, vB : real:
util(A, b, vA) & util(B, b, vB) $\Rightarrow$ pareto_distance(vA, vB) $= \varepsilon$

This bid property states that with respect to agents A and B, the Pareto inefficiency of a bid b is the number ε that indicates the distance to the Pareto Efficient Frontier according to some distance measured in terms of their utilities vA and vB. Here, bids are projected onto points in the plane of utilities. The function to measure the distance in the plane can still be filled in, e.g., the sum of absolute differences of coordinates, or the square root of the sum of squares of the differences, or the maximum of the differences of the coordinates. The Pareto Efficient Frontier is the set of all bids b for which there is no other bid b' that dominates b. Hence, in case the Pareto Inefficiency of a bid is 0, there is no other bid that dominates it. By filling in the resulting agreement of a negotiation for bid b, the property is transformed into a result property. In general, determining the number ε for which this property holds is a good measure for checking the success of the negotiation process. In a similar way, the property nash_inefficiency can be formulated, which calculates the distance from a certain bid to the Nash Point. This is the point (on the Pareto Efficient Frontier) for which the product of both utilities is maximal, see e.g. [21].

4. The SAMIN architecture

SAMIN is a Prolog-based software environment that has been designed at the Vrije Universiteit Amsterdam for the analysis of multi-issue negotiation processes[1]. Section 4.1 describes the role SAMIN can take in an analysis setting of negotiation processes. Next, Section 4.2 presents a top level overview of the SAMIN architecture. Basically, the system consists of three components: an *Acquisition Component*, an *Analysis Component* and a *Presentation Component*. These components are described in more detail in Sections 4.3, 4.4, and 4.5, respectively.

4.1. SAMIN in its environment

The SAMIN system has been designed to work together in interaction with a human analyst and either human or software agent negotiators. As depicted in Fig. 1, the analyst determines the properties that SAMIN is to use in the analysis of negotiation processes. He or she can select (and if necessary adapt) properties from SAMIN's library, or can construct new properties with the help of SAMIN's special

[1]SAMIN can be downloaded from http://www.few.vu.nl/∼wai/samin/

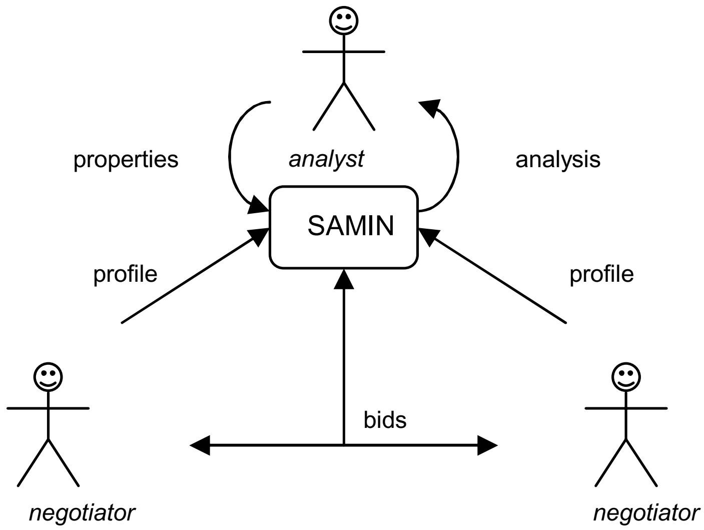

Fig. 1. SAMIN in its environment.

dynamic property editor. SAMIN can only analyse a negotiation process if it has access to the profiles used by the different parties, and the bids exchanged between the parties. SAMIN does not influence the negotiation while it is being carried out, it only observes either during the negotiation, or afterwards.

The analysis result of one or more negotiations is presented to the human analyst. The analyst can use that result for purposes within Cognitive Science (e.g., to analyse human negotiation processes and train human negotiators) or Artificial Intelligence (e.g., to improve the strategies of software agents). Interesting for the future might be to present the results directly after the conclusion of the negotiation to a software agent negotiator that is capable of learning so that the agent can use the result to improve its negotiation skill by itself. A negotiation process can be monitored directly by SAMIN (if the agents allow interfacing), or the negotiation trace can be written to a file and be analysed in hindsight by SAMIN. The current version of SAMIN is developed especially for closed multi-issue one-to-one negotiations, entailing that the only information exchanged between the negotiators are the bids.

The input required by SAMIN (see Fig. 1) consists of properties, profiles, and traces of bids. Its output consists of an analysis that can be presented in a user-friendly format (see Sections 4.4 and 4.5). As mentioned before, SAMIN offers the user both a library of properties to choose from and a dynamic property editor to create new properties. Profiles can be obtained in two ways. Either the negotiator presents a pre-specified profile to SAMIN or the negotiator can use SAMIN's interactive profile editor to create it in SAMIN. Pre-specified profiles have to be in a format recognised by SAMIN. The trace of bids required by SAMIN can be obtained by SAMIN monitoring the bids exchanged between the negotiators during the negotiation process. This only requires the bids to be in a format recognised by SAMIN and the possibility to "overhear" the communication between the negotiators. Another possibility is that the bids exchanged during a negotiation process are stored in a special file. If the bid-traces are in the right format, SAMIN can perform analysis on one or on a combination of such traces after the negotiation has been completed. If the negotiators wish to do so, they can use SAMIN's bid ontology editor to define what a bid should look like, before entering the negotiation phase. Construction of bid ontology and the profiles is part of the pre-negotiation phase [21].

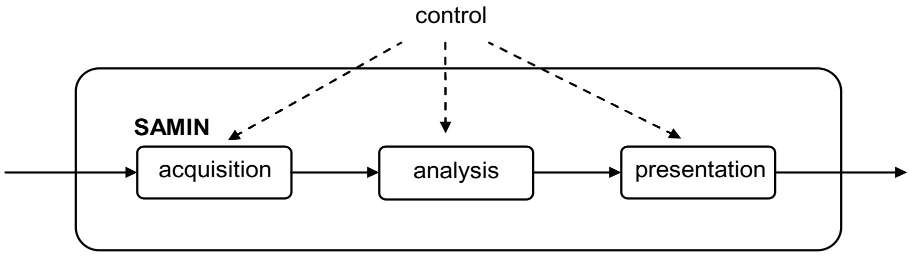

Fig. 2. Global Overview of the SAMIN architecture.

4.2. Top level

At the top level, SAMIN consists of three components: an Acquisition Component, an Analysis Component and a Presentation Component, see Figure 2. Here, the solid arrows indicate data flow. The dotted arrows indicate that each component can be controlled separately by the analyst. The Acquisition Component is used to acquire the input necessary for analysis. The Analysis Component is used to perform the actual analysis (i.e., checking which properties hold for the negotiation process under analysis). Finally, the Presentation Component is used to present the results of the analysis in a user-friendly format. Furthermore, SAMIN maintains a library of properties, templates of properties, bid ontologies, and profile ontologies (not shown in Fig. 2). The working of the three components will be described in detail in the next sections.

4.3. The acquisition component

The acquisition component is used to obtain the required input for the analysis. It consists of an *ontology editor*, a *dynamic property editor* and a *trace determinator*.

The ontology editor is used for the construction of bid ontologies and profile ontologies necessary to automatically interpret the bids exchanged by the negotiators, and to automatically interpret the profiles of the negotiators. The ontology editor is typically used to construct a bid ontology and a profile ontology, thus allowing the user to identify the issues to be negotiated, the values that each of these issues can take, and the structure of bids, in the bid ontology. Furthermore, in specifying the profile ontology the user identifies the possible evaluations that can be given to values, and the utility functions of bids.

The dynamic property editor supports the gradual formalisation of dynamic properties in TTL format. The editor offers a user interface that allows the analyst to construct dynamic properties, represented in a tree-like format.

The trace determinator can be used interactively with the analyst to determine what traces to use in the analysis. The user can interactively locate the files containing the traces to be checked. The traces themselves can be of three categories: (human) empirical traces, simulated traces, and mixed traces. An empirical trace is the result of an existing human negotiation process. A simulated trace is the result of an automated negotiation process. A mixed trace is the result of a human negotiating with a software agent. To support the acquisition of traces of all three types, a dedicated interface has been created for SAMIN.

4.4. The analysis component

The analysis component currently consists of a *logical analyser* that is capable of checking properties against traces. To this end, the tool takes a dynamic property in TTL format and one or more traces as input, and checks whether the dynamic property holds for the traces.

Traces are represented by sets of Prolog facts of the form holds(state(m1, t(2)), a, true) where m1 is the trace name, t(2) time point 2, and a is a state property as introduced in Section 3.1. The above example indicates that state formula a is true in trace m1 at time point 2. The Analysis Component basically uses Prolog rules for the predicate sat that reduce the satisfaction of the temporal formula finally to the satisfaction of atomic state formulae at certain time points, which can be read from the trace representation. Examples of such reduction rules are:

```
sat(and(F,G)) :- sat(F), sat(G).
sat(not(and(F,G))) :- sat(or(not(F), not(G))).
sat(or(F,G)) :- sat(F).
sat(or(F,G)) :- sat(G).
sat(not(or(F,G))) :- sat(and(not(F), not(G))).
```

In addition, if a dynamic property does not hold in a trace, then the software reports the places in the trace where the property failed.

4.5. The presentation component

The presentation component currently includes a tool that visualises the negotiation space in terms of the utilities of both negotiators. This *visualisation tool* plots the bid trajectory in a 2-dimensional plane, see Fig. 3. The utilities are real values that indicate how a particular bid is evaluated by a negotiator. Details about the calculation of utilities are provided in the next Section.

In Fig. 3, the seller's utility of a bid is on the horizontal axis, and the buyer's utility is on the vertical axis. The light area corresponds to the space of possible bids. In this area, each curve is a continuous line, corresponding to a different combination of discrete issues. The specific position on the line is determined by the continuous issue 'price'. Since in this particular domain 4 discrete issues with 5 possible values occur (see next section), there are already 625 ($= 5^4$) curves. In this Figure, the sequences of actual bids made by both buyer (left) and seller (right) are indicated by the dark points that are connected by the two angular lines. The upper-left point indicates the buyer's first bid, and the lower-right point indicates the seller's first bid. The dotted line indicates the Pareto Efficient Frontier according to the profiles of the negotiating agents, and the short dark lines show the distance from each bid to this frontier. The big dot that is plotted on the Pareto Efficient Frontier (on the right) corresponds to the Nash Point. From this picture, it is clear that both negotiators make more and more concessions (their bids converge towards each other). Eventually, they reach a point that does not lie on the Pareto Efficient Frontier, but is rather close to it anyhow.

5. Human multi-issue negotiation experiments

To illustrate the use of analysing human multi-issue negotiation processes, SAMIN has been applied in a case study. As mentioned in Section 4, the analysis component of SAMIN takes traces and formally specified dynamic properties as input and checks whether a property holds for a trace. Using automatic checks of this kind, some of the properties provided in Section 3 have been checked against empirical traces generated by students during practical sessions in multi-issue negotiation. The domain of the case study, a negotiation about second hand cars, is presented in detail in Section 5.1. Section 5.2 describes the setup of the experiments performed in the case study. The analytical results of the acquired traces will be shown in Section 5.3.

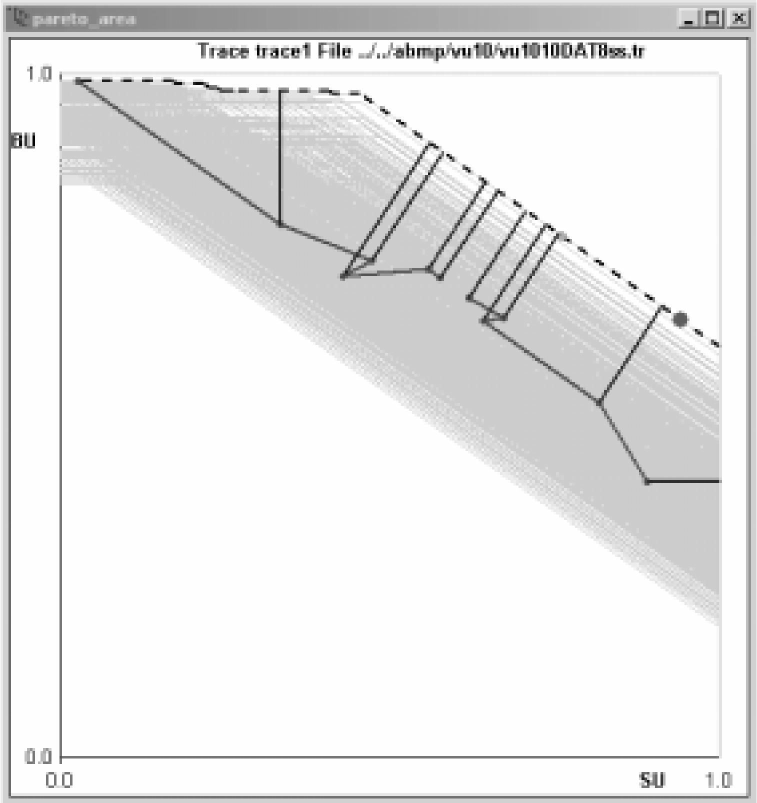

Fig. 3. Visualisation Tool.

5.1. Domain: second hand cars

The protocol used in the experiments is an alternating-offers protocol. In this type of negotiation, a bid has the form of values assigned to a number of issues of the object under negotiation. Here, the object of negotiation is a particular second hand car. Within this domain, the relevant issues are cd_player, extra_speakers, airco, tow_hook and price. Consequently, a bid consists of an indication of which CD player is meant, which extra speakers, airco and tow hook, and what the price of the bid is. The goal of the negotiators is to find agreement upon the values of the four accessories and the price. Here, the price issue has a continuous value, whilst the other four issues have a discrete value from the set {good, fairly_good, standard, meager, none}. These values are assumed to be objective indicators from a consumer organisation, so there can be no discussion about whether a certain CD player is good or fairly good.

Before the negotiation starts, both parties specify their *negotiation profile*: for all issues with discrete values they have to assign a number to each value, indicating how satisfied they would be with that particular value for the issue (e.g. “I would be very happy to buy/sell a good CD player, a bit less happy with a fairly good CD player, ... ” and so on). The buyer also has to indicate what is the maximum amount of money (s)he would be willing to spend. Moreover, both parties have to assign a number to each of the issues, indicating how important they judge that issue (e.g. “I don’t care that much which CD player I will buy/sell”). Notice that this does not conflict with the above statements. An example negotiation profile for a buyer is shown in Fig. 4. In addition to this negotiation profile, the seller is

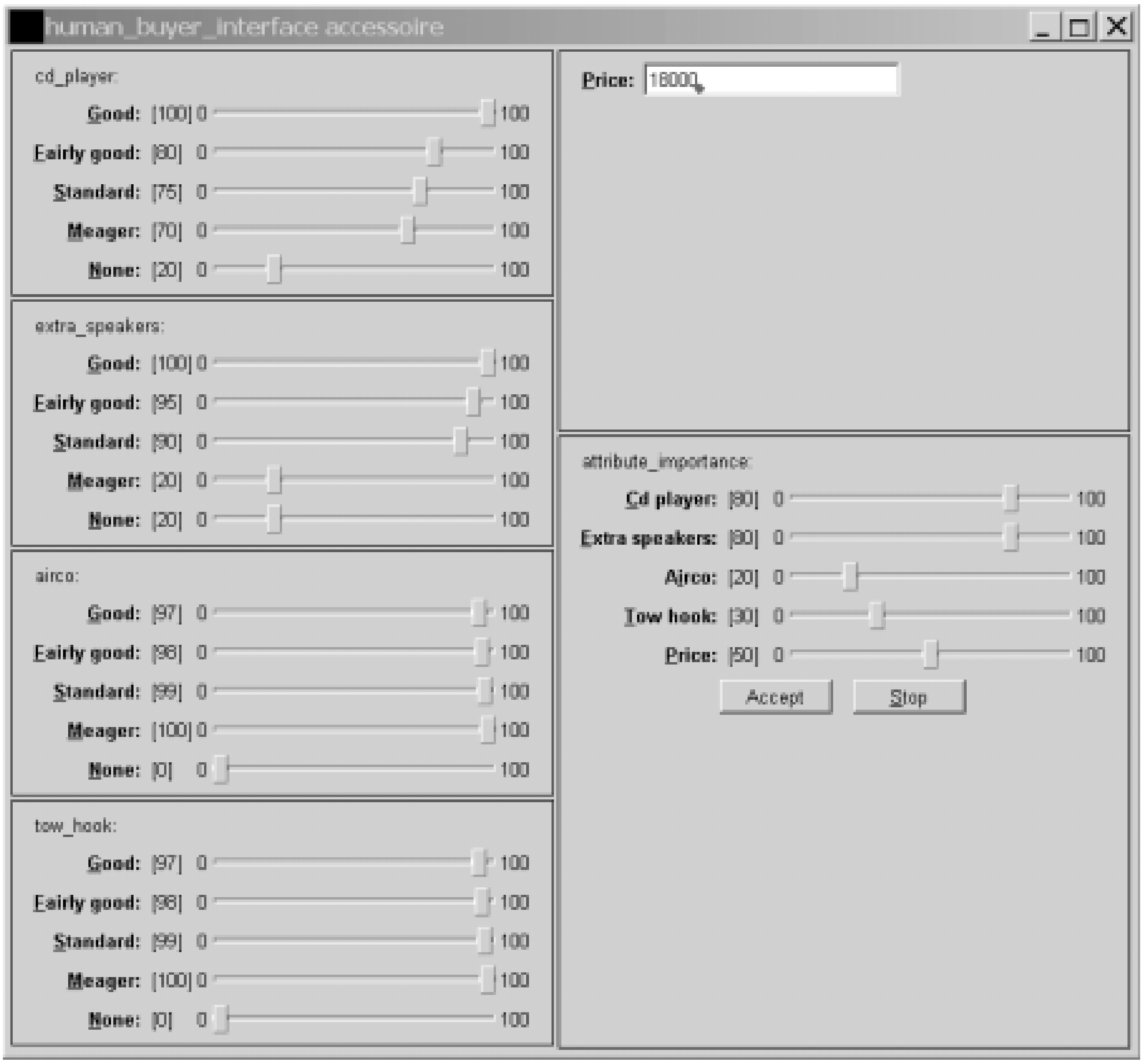

Fig. 4. Example Buyer's Negotiation Profile.

provided with a *financial profile*. This is a list of all issues, where for each issue it is indicated how much it costs, both to buy it and to build it into the car. Since the focus is on closed negotiation, none of the profiles will be available for the other party. However, SAMIN has access to both profiles.

When both parties have completed their profiles, the negotiation starts. To help human negotiators generating their bids, the system offers a special tool that calculates the utility of a bid before it is passed to the opponent.

The utility U_B of a bid B is defined by the weighted sum over the issue evaluation values $E_{B,j}$ for the different issues denoted by: $U_B = \Sigma_j w_j \, E_{B,j}$. The weight factors w_j are based on the issue importance factors. Here scaling takes place (the sum of weight factors is made 1, and the evaluation values $E_{B,j}$ are between 0 and 1) so that the utility is indeed is between 0 and 1; for more details, see [14]. Since

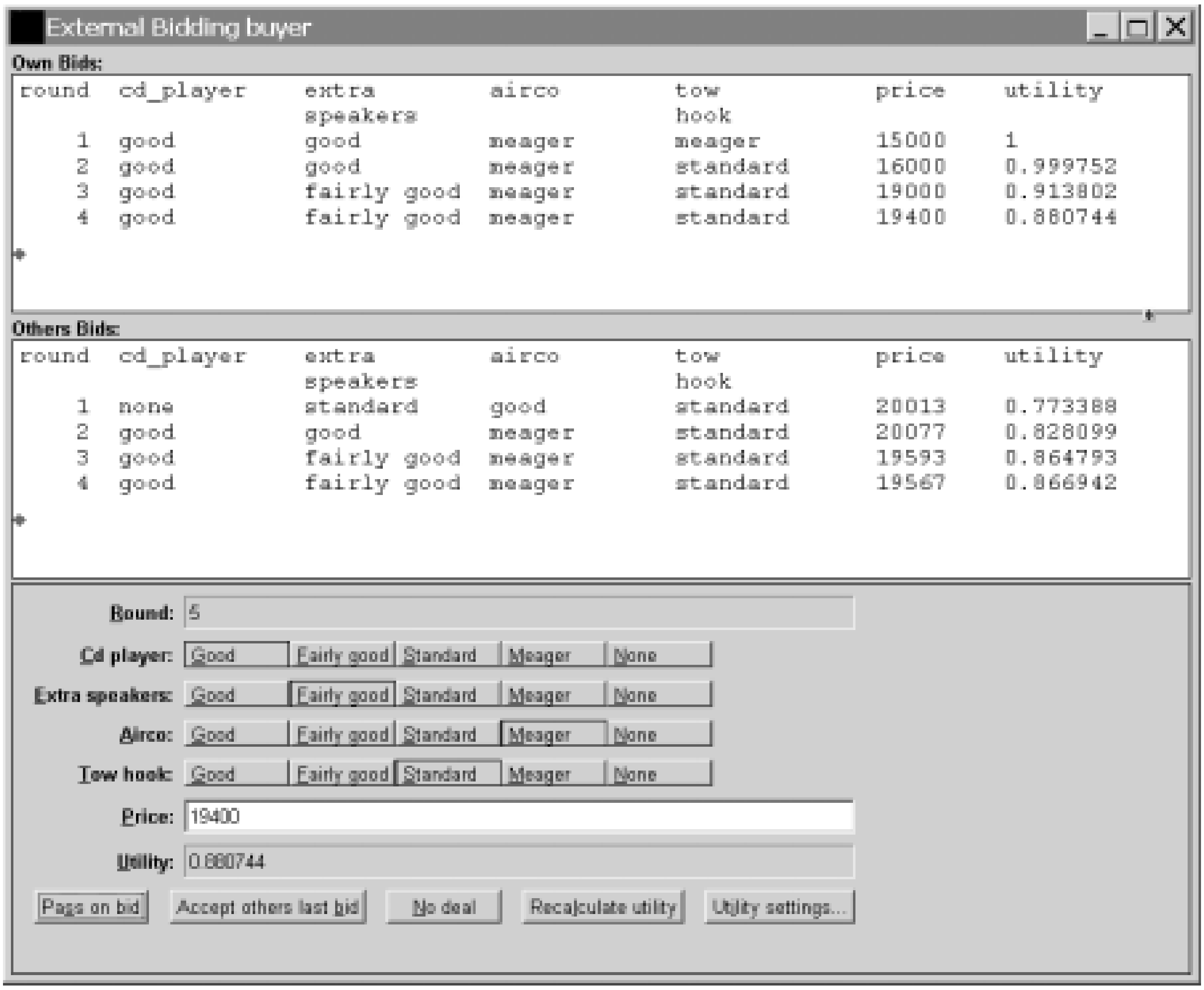

Fig. 5. Example Negotiation Trace.

the negotiators have individual negotiation profiles, for each bid the seller's utility of the bid is different from the buyer's utility of the bid.

Besides for facilitating the bidding process, the profiles are used by SAMIN to analyse the resulting traces. For example, to check whether the property Pareto-Monotony holds (i.e., "For each combination of successive bids b1, b2 in the trace, both agents prefer bid b2 over bid b1"), the software must have a means to determine when an agent "prefers" one bid over another.

5.2. *Experimental setup*

5.2.1. *Participants*

74 subjects participated in the experiment, in three different sessions. All sessions took place during a master class for students of the final classes of the VWO (a particular type of Dutch High School). The age of the students mostly was 17 years, but varied between 14 and 18 years. Most of them were males. In the first session, in March 2002, 30 students participated. In the second, in March 2003, 28 students participated. In the third session, in November 2003, 16 students participated.

5.2.2. Method

Before starting the experiment, the participants were provided some background information on negotiation and in particular about multi-issue negotiation. Some basic negotiation strategies were discussed. In addition, the second hand car example was explained. Then they were asked to start negotiating, thereby taking a profile in mind (that had to be specified first) aiming at obtaining the best possible deal, without showing their own profile to the opponent. The negotiation process was performed using different terminals over a network, which allowed each participant to negotiate with another anonymous participant. All negotiators could input their bids within a special interface. The resulting negotiation traces were logged by the system, so that they could be re-used for the purpose of analysis. A screenshot of an example negotiation trace is depicted in Figure 5. This trace is shown from the perspective of the buyer. In the upper part of the window, the buyer's own bids are displayed, including the buyer's utility for each bid. In the middle part, the bids of the seller are displayed, including the buyer's utility for each bid. The lower part consists of the bidding interface, which allows the buyer to input his bid and pass it to the seller.

5.3. Results of the human experiments

Using the SAMIN prototype, a number of relevant dynamic properties for multi-issue negotiation (also see Section 3) have been checked against the traces that resulted from the experiments. In this section, the most important results are presented.

Obviously, the property bid_alternation (Section 3.1) holds for all traces. This means that all participants have committed to the protocol, which prescribes that as long as the negotiation lasts, a bid from A to B should be followed by a bid from B to A.

In none of the traces, the pareto_inefficiency (Section 3.2) of the resulting deal was equal to 0. In several cases, during the negotiation some bids made by one of both parties temporarily lay on the Pareto Efficient Frontier, but the resulting bids never did. On average, the negotiating agents performed only slightly above halfway, i.e., the resulting bids lay somewhat above the middle of the space of possible bids (the grey area of Fig. 3). Apparently, it is difficult for human negotiators to guess the Pareto Inefficiency. As a result, they find it hard to decide what is the right moment to accept a proposal.

As can be derived from the previous conclusion, also the Nash Point was never reached in any final agreement, nor was it reached during any of the negotiations.

When used as a trace property, the property strict_Pareto_monotony (Section 3.2) did not hold in any of the traces. When used as a limited interval property, it sometimes held during a very short interval, but hardly ever during more than three steps. Apparently, the profiles of the negotiating parties were often strongly opposed, meaning that a gain for one party implies a loss for the other. However, when changing the criterion of strict domination into weak domination, the property often held for larger intervals. Most of the time, these intervals corresponded to the "end phase" of the negotiation: the phase in which the only issue on which no agreement has yet been reached, is the price.

6. Related work

This section discusses the literature on the analysis of negotiation processes. Moreover, it reviews automated negotiation systems that use incomplete information described in the literature and compares them to the work reported here.

In the literature on negotiation a number of systems are described. Sometimes it is stated what properties these systems have, sometimes not. If properties are mentioned they can be of different types, and also the justifications of them can be of different degree or type. This section discusses the literature on properties of negotiation and analytical results of implemented systems and of human case studies.

Faratin, Sierra, and Jennings [6] concentrate on many parties, many-issues, single-encounter closed negotiations with an environment of limited resources (time among them). Agents negotiating using the model are guaranteed to converge on a solution in a number of situations. The authors do not compare the solutions found to fair solutions (Nash Equilibrium, Maximal Social Welfare, Maximal Equitability), nor whether the solutions are Pareto Efficient.

Klein, Faratin, Sayama, and Bar-Yam [16] developed a mediator-based negotiation system to show that conceding early (by both parties) often is the key to achieving good solutions. Hyder, Prietula, and Weingart [12] showed that substantiation (providing rationale for your position to persuade the other person to change their mind) interferes with the discovery of optimal agreements.

Weingart et al. [24] found that the Pareto efficiency of agreements between naïve negotiators could be significantly improved by simply providing negotiators with descriptions of both integrative and distributive tactics. Although Pareto *efficiency* was positively influenced by the tactics, Pareto *optimality* was only minimally affected. Compared to [8,11,12], the properties identified in current paper are geared towards the analysis of the dynamics of the negotiation process, whereas theirs are more oriented towards the negotiation outcome, rationality and use of resources.

In [8], a model for bilateral multi-issue negotiation is presented, where issues are negotiated sequentially. The issue studied is the optimal agenda for such a negotiation under both incomplete information and time constraints. However a central mediator is used and the issues all have continuous values. The effect of time on the negotiation equilibrium is the main feature studied, from both a game-theoretic and empirical perspective. In earlier research [7] a slightly different model is proposed, but the focus of the research is still on time constraints and the effect of deadlines on the agents' strategies. This contrasts with the model presented here, where efficiency of the outcome and not time is the main issue studied. This is because it was found that, due to the cooperativity assumption, a deal is usually reached in maximum 10-15 steps, if the negotiation speed and tolerance parameters are suitably calibrated.

In [2], the SAMIN system is extended by incorporating heuristics for profile guessing, similar to the work in [5] and [6]. Like [5], in [2] the starting point is the perspective of distributed negotiation, which eliminates the need of a central planner. In addition, in the current paper the heuristic approach is taken and agents are modelled that are able to jointly explore the space of possible outcomes with a limited (incomplete) information assumption. In [5], this is done through a trade-off mechanism, in which the agent selects the value of its next offer based on a similarity degree with previous bids of the opponent. In the work reported here, trade-offs are not explicitly modelled, yet the same effect is achieved through the asymmetric concessions mechanism. An advantage of the model in [2] over the model in [5] is that agents are allowed to take into account not only their own weights, but also those of the opponent in order to compute the next bid. In this way agents may exchange partial preference information for those issues for which their owners feel this does not violate their privacy. Also the initial domain information for the issues with discrete ("qualitative") values is different. In [5], this consists of fuzzy values, while in the model presented here it is a partial ordering of issue weights.

7. Conclusions and future work

The contribution of this work consists of a systematic approach to the analysis of the negotiation process. Different types of properties are identified and for each class a number of properties are defined. The System for Analysis of Multi-Issue Negotiation (SAMIN) is presented and applied in two ways: to analyse human negotiation in a case study and to analyse the effectiveness of guessing and limited information exchange as implemented in a number of software agents. SAMIN consists of three components: an Acquisition Component to acquire the input necessary for analysis, an Analysis Component to perform the actual analysis, and a Presentation Component to presents the results of the analysis in a user-friendly format.

The system has proved to be a valuable tool to analyse the dynamics of human-human closed negotiation against a number of dynamic properties. The analysis given here shows that humans find it difficult to guess where the Pareto Efficient Frontier is located, making it difficult for them to accept a proposal. Although humans apparently do not negotiate in a strictly Pareto-monotonous way, when considering larger intervals, a weak monotony can be discovered. Such analysis results can be useful in two different ways: to train human negotiators, or to improve the strategies of software agents.

Currently, SAMIN is being used to analyse the dynamics of humans negotiating against software agents (with and without guessing strategy, also in setting in which limited preference information is shared; e.g. [13]). Future research is to analyse the dynamics of other types of (e.g., more experienced) human negotiators. Furthermore, the system needs to allow heterogeneous agents, so that a competition of negotiating agents can be set up and the results of that competition formally analysed. In the future, SAMIN will be extended with training facilities for human negotiators, allowing the analyst to test the effectiveness of training methods for negotiation. As a simple extension, for example, if a dynamic property checked in a trace turns out to fail, a more detailed analysis can be given of the part(s) of the formula that cause(s) the failure. Finally, it is planned to extend SAMIN to provide feedback to a negotiator who is in the middle of a negotiation process, where SAMIN only has access to the same information as the negotiator.

Appendix A – Dynamic Properties of Negotiation Processes

bid_alternation(γ**:trace**)

Over time the bids of A and B alternate: thus for all two different moments in time t1, t3, that A generated a bid, there is a moment in time t2, with t1 < t2 < t3, such that A received a bid generated by B.

∀ A, B: AGENT, ∀ b1, b3: BID, ∀ t1, t3:
  t1 < t3 &
  state(γ, t1, output(A)) |= to_be_communicated_to_by(b1, B, A) &
  state(γ, t3, output(A)) |= to_be_communicated_to_by(b3, B, A) ⇒
    ∃b2, ∃t2: t1 < t2 < t3 &
      state(γ, t2, input(A)) |= communicated_to_by(b2, A, B)

is_followed_by(γ**:trace, A:AGENT, t1:time, b1:BID, B:AGENT, t2:time, b2:BID**)

In a negotiation process γ bid b1 at time t1 is followed by a bid b2 at time t2 iff bids b1 and b2 are subsequent bids in γ.

state(γ, t1, output(A)) |= to_be_communicated_to_by(b1, A, B) &
state(γ, t2, output(B)) |= to_be_communicated_to_by(b2, B, A) &
t1 < t2 &
[∀t3, ∀C, D: AGENT, ∀b3: BID:
  t1 < t3 < t2 ⇒ state(γ, t3, output(C)) | ≠ to_be_communicated_to_by(b3, C, D)]

agent_consecutively_bids_to(γ**:trace, A:AGENT, t1:time, b1:BID, t2:time, b2:BID, B:AGENT**)
In a negotiation process γ agent A consecutively bids b1 at time t1 and then b2 at time t2 to agent B.
state(γ, t1, output(A)) |= to_be_communicated_to_by(b1, A, B) &
state(γ, t2, output(A)) |= to_be_communicated_to_by(b2, A, B) &
t1 < t2 &
[∀t3, ∀b3: BID: t1 < t3 < t2 ⇒ state(γ, t3, output(A)) | ≠ to_be_communicated_to_by(b3, A, B)]

stop_criterion(γ**:trace, A:AGENT, t2:time**)
The stop criterion holds for agent A at time t, if at time t agent A receives a bid by negotiation partner B that is at least as good as the last bid made by A.
∃t1, ∃B: AGENT, ∃b1, b2: BID:
state(γ, t2, input(A)) |= communicated_to_by(b2, A, B) &
state(γ, t1, output(A)) |= to_be_communicated_to_by(b1, B, A) &
is_followed_by(γ, t1, b1, t2, b2) &
util(γ, A, b1) ≤ util(γ, A, b2)

negotiation_continuation(γ**:trace**)
For both A and B, unless the stop criterion holds, a new proposal is generated by A upon receival of a proposal by B.
∀t, ∀A, B: AGENT, ∀b1: BID:
¬stop_criterion(γ, A, t) &
state(γ, t, input(A)) |= communicated_to_by(b1, A, B) ⇒
[∃b2: BID ∃t2: t2 > t & state(γ, t2, output(A)) |= to_be_communicated_to_by(b2, B, A)]

strictly_dominates(b1:BID, b2:BID, A:AGENT, B:AGENT)
A bid b1 dominates a bid b2 with respect to agents A and B iff both agents prefer bid b1 over bid b2.
∀vA1, vA2, vB1, vB2 : real :
util(A, b1, vA1) & util(A, b2, vA2) & util(B, b1, vB1) & util(B, b2, vB2) ⇒
vA1 > vA2 & vB1 > vB2

weakly_dominates(b1:BID, b2:BID, A:AGENT, B:AGENT)
A bid b1 dominates a bid b2 with respect to agents A and B iff both agents prefer bid b1 over bid b2.
∀vA1, vA2, vB1, vB2: real:
util(A, b1, vA1) & util(A, b2, vA2) & util(B, b1, vB1) & util(B, b2, vB2)
⇒ vA1 ⩾ vA2 & vB1 ⩾ vB2

strictly_better_social_welfare(b1:BID, b2:BID, A:AGENT, B:AGENT)
The social welfare of bid b1 is better than that of bid b2 with respect to agents A and B iff the sum of the utility values of bid b1 is bigger than the sum of the utility values of bid b2. See also [6,10].
∀vA1, vA2, vB1, vB2: real:
util(A, b1, vA1) & util(A, b2, vA2) & util(B, b1, vB1) & util(B, b2, vB2) ⇒ vA1 + vB1 > vA2 + vB2

strictly_better_equitability(b1:BID, b2:BID, A:AGENT, B:AGENT)
A bid b1 has a better equitability than bid b2 with respect to agents A and B iff the difference in the utility values of bid b1 is less than the difference in utility values of bid b2.

∀vA1, vA2, vB1, vB2: real:
util(A, b1, vA1) & util(A, b2, vA2) & util(B, b1, vB1) & util(B, b2, vB2) ⇒ | vA1 - vB1 | < | vA2 - vB2 |

ε-equitability(b:BID, A:AGENT, B:AGENT, ε:real)
A bid b has ε-equitability with respect to agents A and B iff the difference in the utility values of bid b is less than ε. Thus, a bid that has an equitability of 0 has a maximum equitability. This definition corresponds to the idea of Raiffa to maximize the minimum utility [21].
∀vA, vB: real:
util(A, b, vA) & util(B, b, vB) ⇒ | vA - vB | ⩽ ε

pareto_inefficiency(b:BID, A:AGENT, B:AGENT, ε:real)
With respect to agents A and B, the Pareto inefficiency of a bid b is the number ε that indicates the distance to the Pareto Efficient Frontier according to some distance measure in utilities. Here the bids b1 and b2 are viewed as points in the plane of utilities.
∀vA, vB: real:
util(A, b, vA) & util(B, b, vB) ⇒ pareto_distance(vA, vB) =ε

making_global_concession(γ:trace, A:AGENT, t1:time, b1:BID, t2:time, b2:BID, B:AGENT)
In a negotiation process γ agent B makes a global concession to agent B with respect to bid b1 at time t1 and bid b2 at time t2 iff both bids are consecutive, and b2 has a lower utility than b1, from A's perspective. A similar property could be defined stating that an agent receives a global concession from another agent.
agent_consecutively_bids_to(γ, A, t1, b1, t2, b2, B) &
∀vA1, vA2: real:
util(A, b1, vA1) & util(A, b2, vA2) ⇒ vA1 > vA2

configuration_differs(b1:BID, b2:BID)
Two bids b1 and b2 differ in configuration iff there is an issue that has a different value in both bids. Similar properties could be defined stating that two bids differ in configuration in at least x issues.
∃a: ISSUE, ∃v1, v2: VALUE:
value_of(b1, a, v1) & value_of(b2, a, v2) & v1 ≠ v2

agent_views_agent_makes_config_variation(γ:trace, A:AGENT, B:AGENT, t1:time, b1:BID, t2:time, b2:BID)
In the view of agent A, agent B varies the configuration, but not the utility. Note that one agent can both be agent A and B, or A and B can refer to different agents.
agent_consecutively_bids_to(γ, A, t1, b1, t2, b2, B) &
configuration_differs(b1, b2) &
∀vA1, vA2: real:
util(A, b1, vA1) & util(A, b2, vA2) ⇒ vA1 = vA2

agent_views_agent_makes_strict_ε-progression(γ:trace, A:AGENT, B:AGENT, t1:time, b1:BID, t2:time, b2:BID, ε:real)
In the view of agent A, the two consecutive bids b1 and b2 made at times t1 and t2 by agent B show minimum ε-progression in utility iff the second bid is at least ε higher than the first bid. Note that one agent can both be agent A and B, or A and B can refer to different agents.

agent_consecutively_bids_to(γ, A, t1, b1, t2, b2, B) &
$\forall$vA1, vA2: real:
util(A, b1, vA1) & util(A, b2, vA2) $\Rightarrow$ vA2 - vA1 $> \varepsilon$

strict_pareto_monotony(γ:trace, tb:time, te:time)
A negotiation process γ is Strictly Pareto-monotonous for the interval [t1, t2] iff for all subsequent bids b1, b2 in the interval b2 dominates b1:
$\forall$t1, t2, $\forall$A, B: AGENT, $\forall$b1, b2: BID
[tb $\leqslant$ t1 $<$ t2 $\leqslant$ te & is_followed_by(γ, A, t1, b1, B, t2, b2)]
$\Rightarrow$ strictly_dominates(γ, b2, b1, A, B)

weak_pareto_monotony(γ:trace, tb:time, te:time)
A negotiation process γ is Weakly Pareto-monotonous for the interval [t1, t2] iff for all subsequent bids b1, b2 in the interval b2 weakly dominates b1:
$\forall$t1, t2, $\forall$A, B: AGENT, $\forall$b1, b2: BID
[tb $\leqslant$ t1 $<$ t2 $\leqslant$ te & is_followed_by(γ, A, t1, b1, B, t2, b2)]
$\Rightarrow$ weakly_dominates(γ, b2, b1, A, B)

References

[1] T. Bosse, C.M. Jonker, L. van der Meij, A. Sharpanskykh and J. Treur, Specification and Verification of Dynamics in Cognitive Agent Models, in: *Proceedings of the Sixth International Conference on Intelligent Agent Technology, IAT'06*, T. Nishida, et al., eds, IEEE Computer Society Press, 2006, pp. 247–254.

[2] T. Bosse, C.M. Jonker, L. van der Meij, V. Robu and J. Treur, A System for Analysis of Multi-Issue Negotiation, in: *Software Agent-Based Applications, Platforms and Development Kits*, R. Unland, M. Klusch and M. Calisti, eds, Birkhaeuser Publishing Company, 2005, pp. 253–280.

[3] T. Bosse, C.M. Jonker and J. Treur, Formalisation of Dynamic Properties of Multi-Issue Negotiations. Vrije Universiteit Amsterdam, Department of Artificial Intelligence. Technical Report, 2004.

[4] A. Byde and K.-Y. Chen, AutONA: A System for Automated Multiple 1-1 Negotiation, *Fourth ACM Conference on Electronic Commerce*, 198–199.

[5] P. Faratin, C. Sierra and N.R. Jennings, Using Similarity Criteria to Make Issue Trade-offs in Automated Negotiations, *Artificial Intelligence Journal* **142** (2003), 205–237. Also in: *Proceedings of ICMAS-2000*, Boston, MA., 119–126.

[6] P. Faratin, C. Sierra and N.R. Jennings, Negotiation decision functions for autonomous agents, *International Journal of Robotics and Autonomous Systems* **24** (1998), 159–182.

[7] S.S. Fatima, M. Wooldridge and N.R. Jennings, Optimal Agendas for Multi-Issue Negotiation, in *Proceedings of the Second International Conference on Autonomous Agents and Multiagent Systems* (*AAMAS-03*), Melbourne, 2003, pp. 129–136.

[8] S.S. Fatima, M. Wooldridge and N.R. Jennings, Optimal Negotiation Strategies for Agents with Incomplete Information, in *Intelligent Agents VIII*, Springer-Verlag LNAI, (vol. 2333), March 2002, pp. 377–392.

[9] M. Fisher, Temporal Development Methods for Agent-Based Systems, *Journal of Autonomous Agents and Multi-Agent Systems* **10** (2005), 41–66.

[10] A. Galton, Temporal Logic. *Stanford Encyclopedia of Philosophy*, URL: http://plato.stanford.edu/entries/logic-temporal/#2, 2003.

[11] A. Galton, Operators vs Arguments: The Ins and Outs of Reification, *Synthese* **150** (2006), 415–441.

[12] E.B. Hyder, M.J. Prietula and L.R. Weingart, Getting to Best: Efficiency versus Optimality in Negotiation, *Cognitive Science* **24**(2) (2000), 169–204.

[13] C.M. Jonker and V. Robu, Automated multi-attribute negotiation with efficient use of incomplete preference information, *Proceedings of the 3rd International Conference on Autonomous Agents and Multi-Agent Systems* (*AAMAS-04*), New York, July 2004. Extended version included in: C.M. Jonker, V. Robu and J. Treur, An Agent Architecture for Multi-Attribute Negotiation Using Incomplete Preference Information, *Autonomous Agents and Multi-Agent Systems* **27** (2007), 131–152.

[14] C.M. Jonker and J. Treur, An Agent Architecture for Multi-Attribute Negotiation, in: *Proceedings of the 17th International Joint Conference on AI, IJCAI'01*, B. Nebel, ed., 2001, pp. 1195–1201. Extended version included in: C.M. Jonker, V. Robu and J. Treur, An Agent Architecture for Multi-Attribute Negotiation Using Incomplete Preference Information, *Autonomous Agents and Multi-Agent Systems* **27** (2007), 131–152.

[15] C.M. Jonker and J. Treur, Compositional Verification of Multi-Agent Systems: a Formal Analysis of Pro-activeness and Reactiveness, *International Journal of Cooperative Information Systems* **11** (2002), 51–92.

[16] M. Klein, P. Faratin, H. Sayama and Y. Bar-Yam, Negotiating Complex Contracts. Paper 125 of the Center for eBusines@MIT. http://ebusiness.mit.edu, 2001.

[17] R. Kowalczyk and V. Bui, On Constraint-Based Reasoning in e-Negotiation Agents, in: *Agent-Mediated Electronic Commerce III, Current Issues in Agent-Based Electronic Commerce Systems*, F. Dignum and U. Cortés, eds, Lecture Notes in Computer Science, vol. 2003, Springer – Verlag, pp. 31–46.

[18] A.R. Lomuscio, M. Wooldridge and N.R. Jennings, A classification scheme for negotiation in electronic commerce, *International Journal of Group Decision and Negotiation* **12** (2003), 31–56.

[19] D.G. Pruitt, *Negotiation Behavior*, Academic Press, 1981.

[20] H. Raiffa, The art and science of negotiation, *Harvard University Press*, Cambridge, Mass., 1982.

[21] H. Raiffa, *Lectures on Negotiation Analysis*, PON Books, Program on Negotiation at Harvard Law School, 513 Pound Hall, Harvard Law School, Cambridge, Mass. 02138, 1996.

[22] J.S. Rosenschein and G. Zlotkin, *Rules of Encounter: Designing Conventions for Automated Negotiation among Computers*, MIT Press, Cambridge, MA, 1994.

[23] T. Sandholm, Distributed rational decision making, in: *Multi-agent Systems: A Modern Introduction to Distributed Artificial Intelligence*, G. Weiss, MIT Press, 1999, pp. 201–258.

[24] L.R. Weingart, E.H. Hyder and M.J. Prietula, Knowledge matters: The effect of tactical descriptions on negotiation behavior and outcome, *Journal of Applied Psychology* **78** (1996), 504–517.

Authors' Bios

Tibor Bosse is an Assistant Professor at the Department of Artificial Intelligence at the Vrije Universiteit Amsterdam. He obtained his Ph.D. in 2005 at the same university, graduating on a project that addressed the analysis of the dynamics of cognitive processes. Currently, his research activities are focussed on modelling dynamics of agent systems in various disciplines, in particular Biology, Cognitive Science, and Criminology and applied within Ambient Intelligence.

Catholijn Jonker received her Ph.D. degree in Computer Science in 1994 from Utrecht University. From 1995 to 2004 she had a permanent position as Assistant and Associate Professor in the Department of Artificial Intelligence at the Vrije Universiteit Amsterdam. After that for two years she held a position as full professor in AI and Cognitive Science at the Radboud University Nijmegen. Since 2006 she has a position as full professor in Human Computer Interaction and Artificial Intelligence at the Delft University of Technology. Her research has focussed on the design and analysis of agent systems and their applications to information agents and Electronic Commerce. Currently a general theme of her reseach interests is dynamics of behaviour of multiple agents (human and software) in a dynamic environment.

Lourens van der Meij has a background in Physics and Computer Science. In 1976 he graduated at the Molecular Physics Group of Leiden University, where he continued doing research. He studied Theoretical Computer Science at the Delft University of Technology. After graduating in 1989, he started working as a scientific programmer in the Artificial Intelligence Group at the Vrije Universiteit Amsterdam, working for projects related to knowledge-based agent systems. Currently his focus is on Semantic Web Research: he is involved in the STITCH project, the aim of which is to develop methods for semantic integration of heterogeneous collections in the cultural-heritage domain.

Jan Treur received his Ph.D. in Mathematics and Logic in 1976 from Utrecht University. Since 1986 he works in Artificial Intelligence, from 1990 as a full professor and head of the Department of Artificial Intelligence at the Vrije Universiteit Amsterdam. In the 1990s he headed a research programme on component-based design of knowledge-based and agent systems. In the last seven years the research programme focussed on modelling dynamics of agent systems related to other disciplines such as Biology, Cognitive Science, Criminology, Organisation Theory, and Philosophy of Mind, applied in particular within Ambient Intelligence context.